结合评分比例因子及项目属性的协同过滤算法 *

李淑芝[a]，李志军[a]，邓小鸿[b]

(江西理工大学 a.信息工程学院; b.应用科学学院，江西 赣州 341000)

摘　要：针对传统的协同过滤算法存在用户评分矩阵稀疏及未考虑项目属性之间关系的问题，提出了结合评分比例因子及项目属性的协同过滤算法。首先利用评分矩阵得出项目之间的共同与非共同评分用户数量比矩阵，以此增加项目共同评分用户的影响度，减少用户—项目评分矩阵的稀疏性对项目相似度计算带来的误差；然后对项目属性量化得出其对项目相似度的影响权重，提高项目相似度计算的准确性，根据以上两点提出了一种结合评分比例因子及项目属性权重作为项目相似度权重的算法。实验结果表明该算法在召回率和准确率上相比现有的方法分别提高了 5.1%和 4.7%，算法适用于电商类网站的个性化推荐。

关键词：协同过滤；稀疏矩阵；评分比例因子；项目属性

中图分类号：TP301.6　　doi: 10.3969/j.issn.1001-3695.2018.08.0627

Collaborative filtering algorithm combined with score scale factor and item attribute

Li Shuzhi[a], Li Zhijun[a], Deng Xiaohong[b]

(*a. College of Information Engineering, b. College of Applied* Science, *Jiangxi University of Science & Technology, Ganzhou Jiangxi* 341000, *China*)

Abstract: There exists several issues in traditional collaborative filtering algorithms: a) It has the sparsity of user rating matrix; b) It ignores the relationship between item attributes. Considering all these problems, this paper proposed a novel collaborative filtering algorithm combining score ratio factor and item attribute. The algorithm used the scoring matrix to obtain the ratio matrix of common and non-common score users between items. Therefore, it increased the influence degree of the users of the item common score, and reduced the error caused by the sparsity of the user-item scoring matrix on the item similarity calculation. quantifying the item attribute could obtain the weight of the item similarity, and it also improved the accuracy of the item similarity calculation. According to the above two points, an algorithm combining scoring scale factor and item attribute weight as item similarity weight is proposed. Experimental results show that, it improved the recall rate and accuracy of the algorithm by 5.1% and 4.7% respectively compared with the existing methods. The algorithm is suitable for personalized recommendation of e-commerce websites.

Key words: collaborative filtering; sparse matrix; scoring scale factor; item attribute

0　引言

随着电子商务规模的不断扩大，商品个数和种类快速增长，出现了“信息超载”问题。为了解决这些问题，个性化推荐系统应运而生[1~4]。协同过滤推荐算法是目前应用最广泛的个性化推荐算法，其出色的速度和健壮性，在全球互联网领域备受青睐。协同过滤的原理就是利用某种兴趣相投、拥有共同经验群体的喜好来推荐用户感兴趣的信息，个人通过合作的机制给予信息一定程度的回应（如评分）并记录下来以达到过滤的目的，进而帮助别人筛选信息。协同过滤算法可以分为基于用户（user-based collaborative filtering）和基于项目（item-based collaborative filtering）两方面[5~8]，这两方面都属于最近邻协同过滤推荐，通过评分相似的多个最近邻居的评分向用户产生推荐。

但是随着用户（或项目）规模的急剧扩大，数据变得越来越稀疏，协同过滤推荐算法存在着一些缺陷，主要包括矩阵稀疏造成的推荐结果不精确；项目冷启动问题,即如何为新加入系统的用户推荐。针对以上问题，许多研究者从不同角度对协同过滤推荐算法进行了相应的改进和完善。Ha 等人[9]提出将单个用户评价过的项目构建成项目网络图，根据项目网络图中用户已评分的项目预测其对未评分项目的喜好度并按照降序排名形成推荐集，提高了推荐结果的准确性，但是当用户评价的项目数量较少或者新用户没有评价无法构建项目网络图，存在着冷启动问题。Yagci 等人[10]提出利用评分项目中频繁子集来代替传统的项目相似度矩阵，减少了计算复杂度和避免了传统项目相似度矩阵带来的误差，提高了推荐质量，但是支持度阈值需要动态迭代，算法在通用性方面存在不足。Polato 等人[11]提出了一种基于核方法的协同过滤，该算法在为用户推荐时考虑到了项目流行度的长尾分布及用户个人偏好，在准确率、召回率上得到了提高。孔欣欣等人[12]提出以标签权重评分的形式向用户展示推荐结果并作出合理的解释，证明了推荐结果的有效性。于金明等人[13]提出了一种基于评分相似性和结构相似性两部分构成的新的项目相似度量方法，该方法惩罚活跃用户的逆项目频率，考虑到共同评分用户对相似度的影响，有效地提高了推荐结果的准确性，但是该算法未考虑到项目属性之间的关系。Zhou 等人[14]提出了一种基于置信加权偏差模型的在线协同过滤算法，将用户个人兴趣引入到相似度计算公式中，提高了相似度计

收稿日期：2018-08-24；**修回日期：**2018-10-16　**基金项目：**国家自然科学基金资助项目（61762046）

作者简介：李淑芝（1964），女，江西赣州人，教授，主要研究方向为软件工程和信息隐藏（919165178@qq.com）；李志军（1994），男，河南信阳人，硕士研究生，主要研究方向为数据挖掘；邓小鸿（1982），男，湖北天门人，副教授，博士，主要研究方向为网络信息安全.

算的准确性，缓解了数据稀疏问题以及提高了推荐准确性，但是未考虑到用户及项目属性之间的联系。

为解决上述文献计算相似度时未考虑项目属性及项目冷启动问题，本文提出了一种基于项目属性权重和用户评分比例因子作为项目相似性权重的协同过滤算法，克服了传统的项目相似性计算未能考虑项目属性内在联系及矩阵稀疏性带来的不准确问题。算法同时考虑到项目冷启动问题，通过在系统中找出与大多数用户都相似的用户，然后将这些用户选择的物品推荐给新用户的方式，来解决项目冷启动的问题。

1 传统的协同过滤推荐算法

1.1 用户-项目评分数据表建立

假设有用户集合 $U=\{u_1,u_2,u_3,\cdots,u_n\}$，物品集合 $I=\{i_1,i_2,i_3,\cdots,i_m\}$，$S_{ui}$ 表示用户 u 对物品 i 的评分,用户对物品的评分值在 1～5，用户未评分物品在评分数据中对应为“-”。建立的用户—项目评分数据表 S 如表 1 所示。

表 1 用户-项目评分数据表

Table 1 User-item score table

	i_1	i_2	i_3	···	i_m
u_1	s_{11}	s_{12}	s_{13}	···	-
u_2	s_{21}	-	s_{23}	···	s_{2m}
u_3	s_{31}	s_{32}	-	···	s_{3m}
···	···	···	···	···	···
u_n	s_{n1}	s_{n2}	-	···	s_{nm}

1.2 相似度计算方式

相似度的计算是协同过滤推荐算法的关键步骤，最常用的相似度计算方法有余弦相似度、Pearson 相关性以及修正的余弦相似度。在后面的实验中，本文将利用 Pearson 相关性来计算项目之间的相似度。

使用 Pearson 相关系数计算项目 i 和 j 之间的相似度如式(1)所示。

$$sim(i,j)=\frac{\sum_{u\in U_{ij}}(s_{ui}-\bar{s}_i)(s_{uj}-\bar{s}_j)}{\sqrt{\sum_{u\in U_{uv}}(s_{ui}-\bar{s}_i)^2}\sqrt{\sum_{i\in I_{uv}}(s_{uj}-\bar{s}_j)^2}} \tag{1}$$

其中：U_{ij} 表示项目 i 和 j 的共同用户集合；s_{ui} 和 s_{uj} 分别代表用户 u 对项目 i 和 j 的评分值；$\bar{s}_i$ 和 $\bar{s}_j$ 分别代表项目 i、j 的平均评分。

1.3 产生推荐集

推荐集的产生方式分为基于用户和基于物品的两种方式。基于用户是通过对用户未评分项目进行预测评分并降序排名的方式为用户推荐。基于物品产生推荐集的方式如下：

设根据式(1)计算得出物品相似度矩阵为 $M_i^{m\times m}$，用户 u 的评分矩阵为 $M_u^{m\times 1}$，用户 u 对 m 个物品的喜好程度矩阵 $L_{u,m}^{m\times 1}$ 如式(2)所示。

$$L_{u,m}^{m\times 1}=M_i^{m\times m}M_u^{m\times 1} \tag{2}$$

首先去掉式(2)计算结果中用户 u 已经评价过的物品，然后将剩余物品按照用户喜好度大小降序，选择排名在前 top-N 个物品形成推荐集为用户 u 推荐。在后面的实验中，本文将利用基于物品的推荐集产生方式。

2 改进后的协同过滤推荐算法

2.1 改进的物品相似度计算方法

传统的相似度计量方法当同时评价两个物品的共同用户数量非常少时，根据式(1)得到的物品相似度极大，结果并不合理。为了解决这个问题，引入了两个物品的共同评分用户数量与非共同评分用户数量比例因子的概念。用户评分比例因子的定义如式(3)所示。

$$Factor_{i,j}=\begin{cases}0 & N_{i,j}=0\\ \dfrac{N_{i,j}}{N_i+N_j-2\times N_{i,j}} & N_{i,j}\le N_i+N_j-2\times N_{i,j}\\ 1 & N_{i,j}>N_i+N_j-2\times N_{i,j}\end{cases} \tag{3}$$

其中：Factor 是用户评分比例因子,它的值域是[0,1]；N_i 为评价过目标物品 i 的用户数量；N_j 为评价过相似物品 j 的用户数量；$N_{i,j}$ 为同时评价过物品 i、j 的用户数量；$N_i+N_j-2\times N_{i,j}$ 为从评价过物品 i、j 的用户当中去掉同时评价过物品 i、j 的数量。

传统的基于物品的协同过滤算法未能考虑到项目属性内在的联系，为此本文引入了物品属性权重的概念。针对物品的属性，根据各个物品与之最相似的物品属性之间的关系，将物品属性进行量化，赋予物品属性一个权重。物品属性权重量化的步骤如下：

a）将表 1 的用户-项目评分矩阵数据表进行矩阵转置转为项目—用户评分矩阵数据表，根据相似度计算式(1)计算各个物品的最相似物品，最相似物品集合记为 $MI=\{sim(i_1)_{\max},sim(i_2)_{\max},\cdots,sim(i_m)_{\max}\}$,其中 $i_1\sim i_m$ 属于 1.1 中的定义物品集合 I。

b）设物品的属性表如表 2 所示。

表 2 物品属性表

Table 2 Item attribute table

	p_1	p_2	···	p_k
i_1	1	0	…	1
i_2	0	1	…	1
···	…	…	…	…
i_m	1	1	…	0

其中: p_1～ p_k 为物品的 k 个属性，1 表示对应的物品有对应的属性，0 为对应的物品没有该属性。

根据表 2 可求得物品属性相似度，计算物品 i 与 j 的属性相似度方式如式(4)所示。

$$Psim(i,j)=\frac{|PN_i\cap PN_j|}{|PN_i\cup PN_j|} \tag{4}$$

其中：PN_i，PN_j 分别为物品拥有的属性的量，$PN_i\cap PN_j$ 为物品 i 与 j 共同拥有的属性的量，$PN_i\cup PN_j$ 为物品 i 与 j 拥有的所有属性的量。如计算物品 i 与 j 的属性相似度，物品 i 有属性 物品 j 有属性 p_1,p_3,p_4，则计算方式如式(5)所示。

$$Psim(i,j)=\frac{p_1+p_3}{p_1+p_2+p_3+p_4} \tag{5}$$

c）将 1～k 全排列到 p_1～ p_k，赋予每个属性一个权重，再根据式(4)再次计算每个物品与其最相似物品的相似度。其每种排列下各个物品与其最相似的物品的相似度记为 $V(y)_{i,i\max}$。如物品 1 与其最相似的物品 $sim(i_1)_{\max}$ 在第 1 种排列的情况下，计算的相似度记为 $V(1)_{1,1\max}$。其他物品标记方式类似于物品 1，所有物品在第 y 种排列的情况下的相似度的和记为 $CV(y)$。$CV(y)$ 的计算方式如式(6)所示。

$$CV(y)=\sum_{i=1}^{m}V(y)_{i,i\max} \tag{6}$$

d）记在第 $h(1\le h\le k!)$ 排列情况下最大值为 $CV(h)_{\max}$。$CV(h)_{\max}$ 的计算如式(7)所示。

$$CV(h)_{\max}=\max\{CV(1),CV(2),CV(3),\cdots,CV(k!)\} \tag{7}$$

在第 h 种全排列的情况下，相似度的和取得最大值，大多数物品与其相似的物品都取得最大值，即此种排列情况下

$p_1 \sim p_k$ 为求得的用户属性权重的最优解,此时的推荐更加符合实际情况。

e）根据 d）求得的属性权重，可得物品属性相似度的计算如式(8)所示。

$$MPsim(i,j)=\frac{\sum_{p\in P_i\cap P_j} w_p}{\sum_{p\in P_i\cup P_j} w_p} \tag{8}$$

其中 w_p 代表物品的某一种属性权重。

最后根据引入的用户评分比例因子和物品属性权重，得出最后的物品相似度改进如式(9)所示。

$$EIsim(i,j)=Factor_{i,j}\times MPsim(i,j)\times Peasim(i,j) \tag{9}$$

其中 $Peasim(i,j)$ 为利用式(1)Pearson 相关系数计算得到的物品相似度， $EIsim(i,j)$ 为最终得到的改进的物品相似度计算式。

改进之后的算法首先需要计算 n 个项目之间的相似度，然后又需要在维度为 m 的用户评分向量上计算，所以时间复杂度为 O(n×n×m)，由于 m 和 n 数量级相同，所以最终时间复杂度为 $O(n^3)$。

2.2　项目冷启动解决方法

针对项目的冷启动问题，本文引入了每个用户的相似用户的交集概念，具体的定义如下所示：设用户集合 $U=\{u_1,u_2,u_3,\cdots,u_n\}$，采用式(1)计算用户 i 的前 k 个最相似用户集合为 $U_{i,k}=\{u_{i1},u_{i2},u_{i3},\cdots,u_{ik}\}$，相似用户集合所组成的集合为 $U_{nk}=\{U_{1,k},U_{2,k},U_{3,k},\cdots,U_{n,k}\}$， 记在 U_{nk} 中用户出现次数在 $[n/2,n]$ 的集合为 $Union=\{union_1,union_2,\cdots,union_l\}$,将 $Union$ 集合当中 L 个用户选择的物品按照类别相加并按照次数或者总评分降序结果推荐给系统新加入的用户。

2.3　算法实现

本文改进的算法流程如图 1 所示。

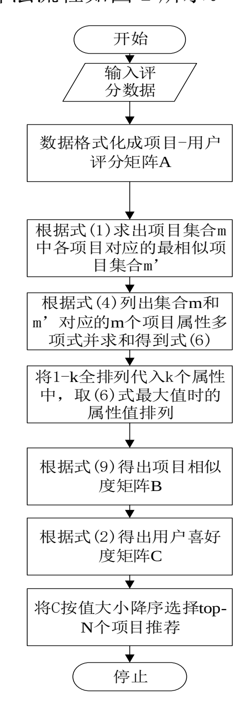


图 1　算法流程

Fig. 1　Algorithm flow

算法　结合评分比例因子及项目属性的协同过滤算法

输入：项目-用户评分矩阵 A，用户-项目评分矩阵 B，用户集合 U，项目集合 I，推荐项目个数 k。

输出：top-N 集合。

a)利用式(1)Pearson 相关系数计算项目相似度。

b)利用式(8)计算物品属性相似度。

c)评分比例因子计算:

```
factor=0;     //评分比例因子
    for S_i,S_j∈A do
//项目共同评分用户数量为 0
    if CO-USER(S_i,S_j)=0 then
        factor=0;
    //共同评分用户数量≤非共同评分用户数量
 else if CO-USER(S_i,S_j) ≤Num|S_i|+Num|S_j|-2*
        CO-USER(S_i,S_j) then
     factor= CO-USER(S_i,S_j)/Num|S_i|+
                    Num|S_j|-2*CO-USER(S_i,S_j);
    else
        factor=1;
    end if
end for
```

d)top-N 集合选择。

```
    top-N= Ø;       //推荐集初始为空
    C[0.. I.length][0.. I.length];  //物品相似度矩阵
    for i∈I do
        for j∈I do
    //项目相似度=Pearson 系数×属性相似度×评分比例因子
 C[i_index][j_index]=getPearson(i,j)*getProperty(i,j)*
                    getfactor(i,j);
        end for
     end for
//计算每个用户对项目喜好程度
for u∈U do
   result[0.. I.length]; //用户对所有项目的喜好程度
     u=B(:,u_score);     //取出 B 中用户 u 的评分
     //喜好度=项目相似度矩阵·用户评分矩阵
     result=Cu;
     //将用户喜好度降序选择排名前 k 个项目
top-N=sorted(result.value, reverse=True, num=k);
end for
```

3　算法仿真实验

3.1　实验环境

1）实验数据集及平台

实验采用经典的协同过滤推荐算法使用的 MovieLens[15] 和 Jester 两个数据集。MovieLens 数据集中包含了 943 个用户的个人信息，如年龄、性别、职业等；1628 部电影的基本信息；100000 个用户对电影的评分，评分范围为 1～5，数据集的评分稀疏性为 1-100000/(943×1682)=93.7%。Jester 数据集包含了 24983 个用户对 100 个笑话的评分，评分总数量为 1761439 个，评分范围为[-10,10]，数据集的稀疏性为 1-1761439/(24983×100)=29.5%。实验是在 Windows10 操作系统下，基于 Python3.6.4 的环境下完成的。

2）实验评价标准

推荐系统的好坏评价标准主要有统计精度度量和决策支持精度度量方法两类，统计精度度量方法中常用的是平均绝对偏差 MAE(mean absolute error)；决策支持精度度量方法中主要有召回率 (recall)、准确率 (precision) 以及 F1-measure[16~18]。本文的评价标准采用准确率和召回率来评估实验结果。

本文假设用户 u_i 在测试集喜欢的项目集合为 T_i，系统推荐的物品集合为 N_i，则针对用户 u_i 的推荐召回率和准确率的

计算方式分别如式(10)(11)所示。

$$P(u_i)=\frac{T_i\cap N_i}{N_i} \tag{10}$$

$$R(u_i)=\frac{T_i\cap N_i}{T_i} \tag{11}$$

整个系统的准确率和召回率是计算用户集合中的每个用户的准确率和召回率，然后取平均值。

3.2　实验结果及分析

为了验证本文算法的有效性，对 MovieLens 和 Jester 数据集的训练集和测试集的数据进行训练测试，在推荐物品个数 top-N 上取值为{10，15，20，25，30，40，50}。实验分为两部分，分别对只引入用户评分比例因子的结果分析和引入评分比例因子及项目属性结合的结果分析。实验与文献[13]提出的算法、Pearson 相关系数、基于 Sigmoid 函数相关性改进的算法[19]及基于 Jaccard 系数改进的算法[20]对比。

实验 1　用户评分比例因子有效性分析

首先利用召回率和准确率来验证式(3)引进用户评分比例因子改进项目相似度的方法，采用上述的 top-N 集合验证。MovieLens 数据集下得到的实验结果如图 2、3 所示。

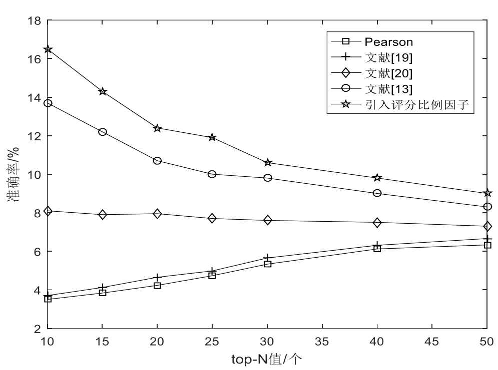


图 2　MovieLens 数据集下准确率比较

Fig. 2　Comparison of precision under movielens dataset

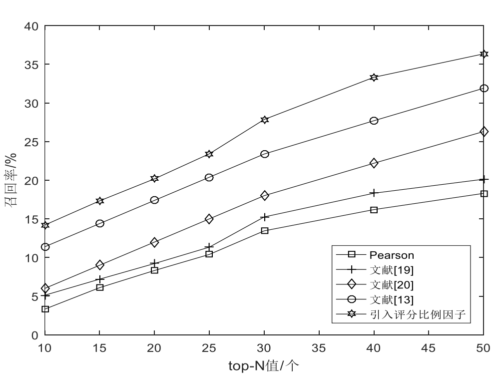


图 3　MovieLens 数据集下召回率比较

Fig. 3　Comparison of recall under movielens dataset

在 Jester 数据集下得到的实验结果如图 4、5 所示。

从图 2 和 4 可以看出，在 Jester 和 MovieLens 数据集下，当 top-N 取值在[10,50]时，引入用户评分比例因子改进之后的准确率高于 Pearson 系数、文献[13,19,20]。改进后的曲线呈现下降趋势是由于在 top-N 取值较小的情况下，如测试当中的 10，就能达到较高的准确率，导致式(10)的分母在变大，分子几乎不变，所以准确率越来越低。改进之后在 top-N 取值为 10 的时候，准确率比当前改进较好的算法文献[13]提高了 2.8%。

在引入用户评分比例因子后，考虑到了项目的共同评分用户对计算项目相似度的影响，当项目共同评分用户与非共同评分用户数量比越大时，说明两个项目之间的相似度越大，从而避免了只依靠用户评分大小来计算相似度带来的不精确问题，提高了项目相似度计算的准确性。

从图 3 和 5 中可以看出，当 top-N 取值在[10,50]的情况下，召回率不断地增加，而且引入用户评分比例因子改进之后的得到的召回率大于 Pearson 系数、文献[13,19,20]算法改进得到的召回率。根据图 2~5 分析，可得到引入用户评分比例因子是有效的。

实验 2　项目属性及评分比例因子结合分析

然后在 Jester 和 MovieLens 数据集下，利用召回率和准确率来验证式(9)结合项目属性及评分比例因子改进之后的项目相似度计算方法是否正确，采用上述的 top-N 集合验证。在 MovieLens 数据集得到的准确率和召回率实验结果如图 6、7 所示。

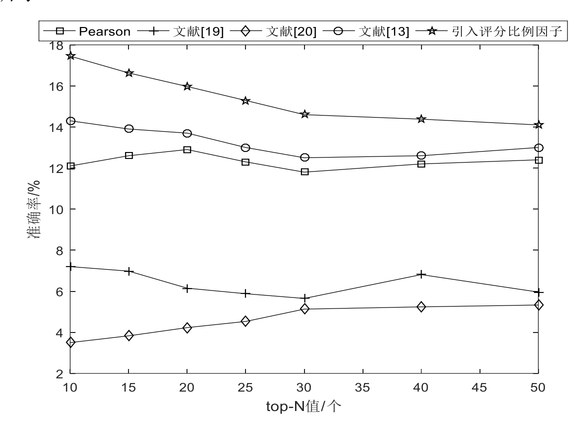


图 4　Jester 数据集下准确率比较

Fig. 4　Comparison of precision under Jester dataset

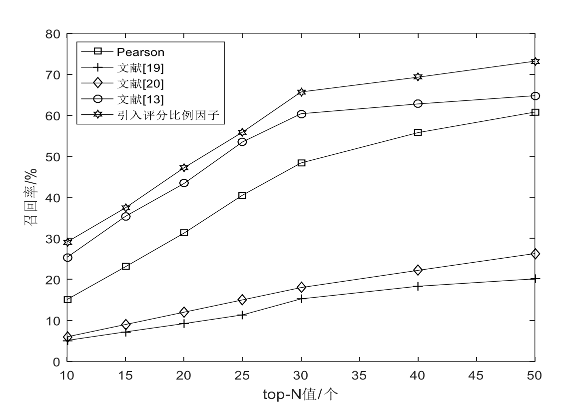


图 5　Jester 数据集下召回率比较

Fig. 5　Comparison of recall under Jester dataset

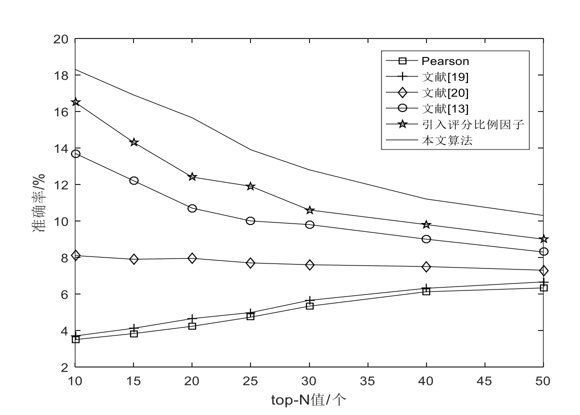


图 6　MovieLens 数据集下准确率比较

Fig. 6　Comparison of precision under movielens dataset

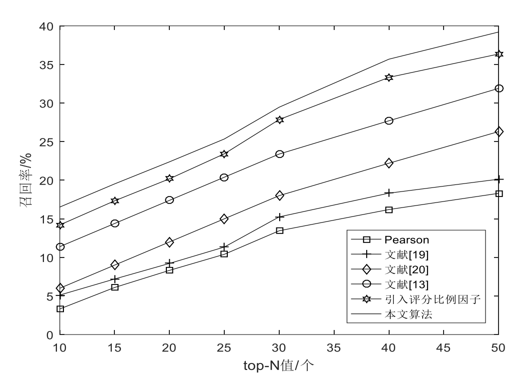


图 7　MovieLens 数据集下召回率比较

Fig. 7　Comparison of recall under movielens dataset

Jester 数据集下得到的实验结果如图 8、9 所示。

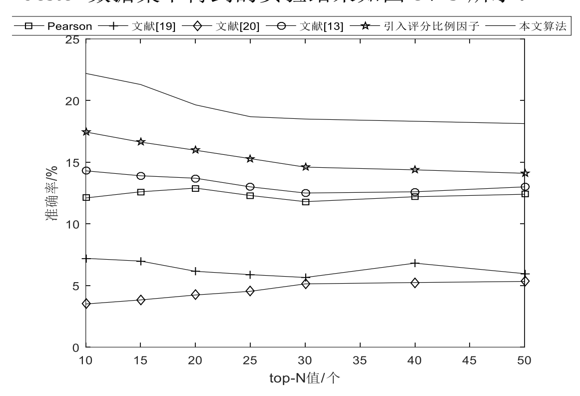


图 8　Jester 数据集下准确率比较

Fig. 8　Comparison of precision under Jester dataset

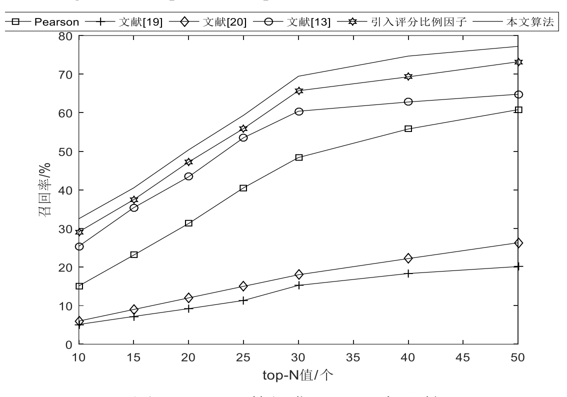


图 9　Jester 数据集下召回率比较

Fig. 9　Comparison of recall under Jester dataset

通过图 6 和 8 中可以看出，在 Jester 和 MovieLens 数据集下，结合评分比例因子及项目属性的改进方式在 top-N 取值较小的情况下就能推荐出用户喜欢的物品，相比与只引入用户评分比例因子的改进，在 MovieLens 数据集下，推荐的准确性提高了 1.9%，在 Jester 数据集下，提高了约 4.7%。图 6 和 8 中，本文方法在 top-N 取值[10，50]的情况下，呈现下降趋势的原因同实验 1 引入用户评分比例因子准确率下降的原因一样。

式(9)在式(3)的基础上，又引入了项目属性之间相似度作为项目相似度权重，提高了项目相似度计算的准确性，降低了仅使用稀疏评分矩阵计算相似度造成的误差，提高了推荐结果的准确性。

从图 7 和 9 中可以看出，当 top-N 取值在[10，50]的情况下，同样，式(9)与 (7)、文献[13，19，20]及传统的 item-CF 相比，在召回率方面都更高一点。由于推荐出的物品是用户喜欢的物品越来越多，即式(11)的分子不断地增加，所以召回率曲线呈上升趋势。说明在推荐数量相同的情况下，结合评分比例因子及项目属性改进之后的推荐效果更佳。

综合实验 1 和 2 可得出，本文提出的结合评分比例因子及项目属性的协同过滤算法考虑到了项目属性之间的内在关系以及同一个项目共同评分用户数量与非共同评分用户数量之间的关系，减少了仅使用评分矩阵带来的误差，使得推荐精度得到了提高，并且在较少的推荐集下就能取得较高的准确率，符合实际的推荐系统需求。

4　结束语

为解决传统的基于项目的协同过滤算法仅使用物品的同现矩阵与用户评分矩阵的乘积作为用户对物品的最终喜好程度、未考虑热门物品带来的影响以及物品内在属性之间的联系，本文提出了结合评分比例因子及项目属性的协同过滤算法。利用用户评分比例因子、物品属性权重以及 Pearson 相关系数代替物品同现矩阵来计算物品相似度。实验结果表明，本文提出的算法在召回率以及准确率上较现有的协同过滤推荐算法得到了提高，证明了本文算法的正确性。未来考虑对结合评分比例因子及项目属性的协同过滤算法在推荐速度和准确性上作进一步的优化。

参考文献：

[1] Hoic-Bozic N, Dlab M H, Mornar V.Recommender system and Web 2. 0 tools to enhance a blended learning model [J]. IEEE Trans on Education, 2016, 59(1): 39-44.

[2] Aguilar J, Valdiviezo-Díaz P, Riofrio G. A General framework for intelligent recommender systems [J]. Applied Computing & Informatics, 2016, 13(2): 147-160.

[3] Liu Ang, Lu Stephen, Zhang Zhinan, *et al.* Function recommender system for product planning and design [J]. CIRP Annals-Manufacturing Technology, 2017, 66(1) 181-184.

[4] Sun Zhoubao, Han Lixin, Huang Wenliang, *et al.* Recommender systems based on social networks [J]. Journal of Systems & Software, 2015, 99 (1): 109-119.

[5] Sun Ping, Li Zhengyu, Han Ziyang, *et al.* An overview of collaborative filtering recommendation algorithm [J]. Advanced Materials Research, 2013, 756-759: 3899-3903.

[6] Shi Yue, Larson M, Hanjalic A. Collaborative filtering beyond the user-item matrix: a survey of the state of the art and future challenges [J]. ACM Computing Surveys, 2014, 47(1): 1-45.

[7] Elahi M, Braunhofer M, Ricci F, *et al.* Personality-based active learning for collaborative filtering recommender systems[C]//Advances in Artificial Intelligence. Cham:Springer International Publishing, 2013: 360-371.

[8] Kurdija A S, Silic M, Vladimir K, *et al.* Efficient global correlation measures for a collaborative filtering dataset [J]. Knowledge-based systems, 2018, 147(3): 36-42.

[9] Ha Taehyun, Lee Sangwon. Item-network-based collaborative filtering: A personalized recommendation method based on a user's item network [J]. Information Processing and Management, 2017, 53(5): 1171-1184.

[10] Yagci A M, Aytekin T, Gurgen F S. Scalable and adaptive collaborative filtering by mining frequent item co-occurrences in a user feedback stream [J]. Engineering Applications of Artificial Intelligence, 2017, 58 (1): 171-184.

[11] Polato M, Aiolli F. Exploiting sparsity to build efficient kernel based collaborative filtering for top-N item recommendation [J].

Neurocomputing, 2016, 268 (6) 17-26.

[12] 孔欣欣，苏本昌，王宏志,等. 基于标签权重评分的推荐模型及算法研究 [J]. 计算机学报，2017, 40(6): 1440-1452. (Kong Xinxin, Su Benchang, Wang Hongzhi, *et al*. Research on the modeling and related algorithms of label-weight rating based recommendation system [J]. Chinese Journal of Computers, 2017, 40(6): 1440-1452.)

[13] 于金明，孟军，吴秋峰. 基于改进相似性度量的项目协同过滤推荐算法 [J]. 计算机应用，2017, 37(5):1387-1391, 1406. (Yu Jinming, Meng Jun, Wu Qiufeng. Item collaborative filtering recommendation algorithm based on improved similarity measure [J]. Journal of Computer Applications, 2017, 37(5): 1387-1391, 1406.)

[14] Zhou Xiuze, Shu Weibo, Lin Fan, *et al*. Confidence-weighted bias model for online collaborative filtering [J]. Applied Soft Computing, 2017, 17(4): 1-12.

[15] Harper F M, Konstan J A. The MovieLens datasets [J]. ACM Trans on Interactive Intelligent Systems, 2016, 5(4): 1-19.

[16] 朱郁筱，吕琳媛. 推荐系统评价指标综述 [J]. 电子科技大学报，2012, 41(2): 163-175. (Zhu Yuxiao, Lyu Linyuan. Evaluation metrics for recommender [J]. Journal of Universty of Electronic Science and Technology of China, 2012, 41(2): 163-175.)

[17] 肖文强，姚世军，吴善明. 一种改进的 top-N 协同过滤推荐算法 [J]. 计算机应用研究, 2018, 35(1): 105-108, 112. (Xiao Wenqiang, Yao Shijun, Wu Shanming. Improved top-N collaborative filtering recommendation algorithm [J]. Application Research of Computer, 2018, 35(1): 105-108, 112.)

[18] Zhang Feng, Gong Ti, Zhao Gansen, *et al*. Fast algorithms to evaluate collaborative filtering recommender systems[J]. Knowledge-Based Systems, 2016, 96(2): 96-103.

[19] Yang Chong, Yu Xiaohui, Liu Yang, *et al*. Collaborative filtering with weighted opinion aspects [J]. Neurocomputing, 2016, 210(5): 185-196.

[20] Hernando A, Ortega F. Collaborative filtering based on significances [J]. Information Sciences, 2016, 185(1): 1-17.